Universidade de Brasília — Faculdade UnB Gama

Disciplina: Estruturas de Dados e Algoritmos Turma: B
Prof.: Fernando W Cruz / Período: 2o./2016 15/09/2016
Aluno: _____ Matrícula: _____

DIÁRIO DE BORDO

1. Em um sistema distribuído os processos possuem um identificador numérico. No processo de inicialização do sistema, são carregados *M* processos, que são armazenados em uma lista encadeada e cada processo é delegado a um dos terminais de processamento. A medida que cada terminal termina o processo que foi a ele atribuído, ele envia uma mensagem ao sistema, que contém o identificador do processo terminado e o caractere *t*. Ao longo do processamento, um terminal pode exigir a criação de novos processos a serem adicionados à lista encadeada de processos. Neste caso, a mensagem conterá um novo identificador, seguido do caractere *n*. As operações de remoção e inserção de processos na lista encadeada do sistema seguem as seguintes regras:
   - A inserção é feita sempre ao final da lista, após o último processo pendente
   - A remoção é feita através da troca dos identificadores do processo a ser removido com o do processo que ocupa a última posição válida da lista encadeada. Após a permuta, o último elemento é removido
   - A lista encadeada do sistema tem uma capacidade máxima de N processos pendentes. Caso uma inserção tente ser realizada com a lista completamente cheia, o processo a ser inserido deve ser descartado
   - Também deverá ser ignorada uma tentativa de remoção quando a lista de processos pendentes estiver vazia ou o identificador do processo fornecido não estiver presente na lista.

Neste contexto, dada a capacidade máxima da lista, o seu estado inicial (do elemento da posição 0 ao elemento da posição M − 1), a quantidade de requisições de inserção e remoção e a descrição de cada requisição, determine o estado final da lista encadeada.

Entrada: A entrada consiste em uma única linha com a capacidade máxima N da lista, o número M de processos a serem carregados na inicialização do sistema, seguidos pelos seus respectivos identificadores, o número R de requisições de inserção e remoção e as descrições das mensagens de inserção e remoção. Todos os valores são separados por espaços em branco, N, M e R são inteiros positivos e a entrada termina com uma quebra de linhas.

Saída: A saída do programa deverá conter a mensagem "V = [$id_1$, $id_2$, . . . , $id_T$ ]", onde $id_i$ é o identificador do *i*-ésimo processo na lista encadeada. Cada identificador deve ser seguido de uma vírgula e um espaço em branco, exceto o último, e todos eles devem estar entre colchetes. Ao final da mensagem deve ser impressa uma quebra de linhas.

| Entradas | Saídas |
|---|---|
| 10 2 1 2 1 3 n | V = [1, 2, 3] |
| 10 3 3 1 2 3 3 t 4 n 5 n | V = [2, 1, 4, 5] |
| 4 3 5 1 8 4 2 n 4 n 6 n 1 t | V = [5, 2, 8] |
| 4 2 5 3 3 5 t 3 t 1 t | V = [] |

2. Todos os semestres ingressam N alunos na disciplina de EDA. Organize um programa em linguagem C para imprimir a relação de alunos e notas obtidas no semestre para essa disciplina, considerando os requisitos abaixo. A impressão deve seguir o seguinte formato:

UnB - Universidade de Brasília / FGA-Gama
EDA - Estruturas de Dados e Algoritmos

>> DESEMPENHO DOS ALUNOS NO SEMESTRE (ORDENADO PELA MÉDIA) <<

| MATRÍCULA | ALUNO | P1 | P2 | P3 | MÉDIA | RESULTADO |
|---|---|---|---|---|---|---|
| 05/00894 | Carlos Alberto da Fonseca | 94 | 96 | 95 | 95 | APR |
| 09/10100 | Zélia Cardoso Senna Aires | 91 | 92 | 93 | 92 | APR |
| 10/65430 | Antônia Maria Nogueira | 79 | 77 | 78 | 78 | APR |
| 08/70000 | Júlia Beltrão | 76 | 74 | 75 | 75 | APR |
| | . | | | | . | |
| | . | | | | . | |
| | . | | | | . | |
| 04/01099 | Bruno Castanheiras | 52 | 56 | 54 | 54 | APR |

Para atender ao que foi solicitado, o programa deve ainda ser composto pelo seguinte grupo de funções:

Uma *procedure* para ler os nomes à partir do teclado. Essa procedure tem o seguinte cabeçalho:

```
void leia_string(char *s) { }
```

Uma função para ler as notas das provas p1, p2 e p3, que podem ir de zero a 100 (números inteiros).

Uma função para calcular a média (float) de cada aluno em função das notas p1, p2 e p3 e o resultado (APR para alunos com média maior ou igual a 50; REP para alunos que não alcançaram essa média).

Uma *procedure* para ordenar os alunos à partir da maior para a menor média obtida na disciplina.

Uma função para imprimir as informações do relatório.

Requisitos:

- A informações devem ser armazenadas em uma lista circular, com cabeçalho, de N nós (informados pelo usuário; além disso, o usuário deve informar mátricula, nome e notas dos N alunos)
- Cada nó da lista encadeada contém, no mínimo, as seguintes informações (observar os tipos de dados para cada campo):

```
struct aluno {
        char   *matricula; /* máximo 10 posições */
        char   *nome;  /* máximo 30 posições */
        int    p1,    p2,    p3;
        float  media;
        char   result[3];
        /* outros campos p/ gestão da lista encadeada */
};
```

- Para o item *(a)*, usar o comando *getchar()* para leitura dos nomes
- Usar identação adequada para os comandos
- Gerar documentação adequada, ou seja, colocar a identificação do aluno (matrícula e nome) e colocar comentários que descrevam o que cada função faz.
- Programas não compiláveis ou que não atendam os critérios estabelecidos não serão aceitos para entrega (embora não precism ser testados com o ejudge)